Cybersecurity Infrastructure Assessment & Implementation - CloudShield team

---

Client Information

Company Name: Titan Industrial Solutions
Point of Contact: James Reynolds, Chief Security Officer
Email: james.reynolds@titanindustrialsolutions.site
Date: 03-06-2025
Project Title: Cybersecurity Infrastructure Assessment & Implementation - CloudShield Team

---

1. Executive Summary

On March 6, Titan Industrial Solutions contacted CloudShield team with the concern of improving its cloud security and preventing cyber attacks.

With the increasing of cybersecurity threats it has become crucial to companies all around the world to protect their infrastructures and secure their data and digital assets. For that reason, organizations like the European Union (EU), drafted a new data privacy and security laws so that companies could work with an international and legal framework. CloudShield team also developed this project according with the AWS framework guidelines.

---

2. Project Scope & Objectives

2.1 Project Scope

CloudShield team was contacted by the Chief Security Officer, James Reynolds, of the Titan Industrial Solutions company, with the purpose of enhancing their cloud security to protect

against cyber threats, industrial espionage and unauthorized access attempts. With this purpose in hands, CloudShield team developed an AWS infrastructure that met the client's requirements.

The company also requested to build this infrastructure in compliance with the NIST cybersecurity guidelines. Given the fact that the client is located in Europe, our team was also obliged to maintain the GDPR framework within the requested documentation, such as the incident response plan and risk assessment report.

2.2 Objectives

- Hardening the AWS-hosted Windows Server DC;
- Implementing a Linux-based data server with encrypted storage for IoT telemetry data;
- Deploying a SIEM solution for detecting anomalies and unauthorized access attempts;
- Conducting a simulated security breach where an attacker attempts to manipulate IoT sensor data (triggering alerts and automated responses);
- Implementing AWS Lambda functions that automate responses to detected threats, such as blocking malicious activity

---

3. Cloud Security Architecture

3.1 AWS Infrastructure Overview

In order to implement this infrastructure, we used the following AWS components and services:

VPC (Virtual Private Cloud)

VPC is an isolated virtual network resource that resembles a traditional network, allowing users to connect with different applications. Within the VPC, the following features were also configured:

- Subnets (1 private, 1 public), where we configured the range of the IP addresses (10.1.2.0 - 10.1.2.127);
- Route table, where it was determined the network traffic direction;
- Gateway, to connect the machine to the internet;

- Elastic IP, to create a mask and redirect an instance failure to another instance;
- NAT (Network Address Translation) gateway, that allows for all of the VPC resources to securely access external subnets (including internet);
- EC2 (Amazon Elastic Compute Cloud) instances, an AWS web service that provides compute capacity. For this project, we configured 2 instances, 1 to for the Windows Server DC and 1 for the Linux-based data server;
- VPC flow logs, a resource that captures IP traffic going to and from the VPC;
- OpenVPN (endpoint/server), an open resource that provides secure web access. This resource was configured in order to be compliant with the GDPR framework and establish an encryption protocol, thus ensuring data-in-transit encryption;
- EBS (Elastic Block Store), a block-level storage solution that allowed us to provide with data-at-rest encryption;
- CloudWatch, a resource that helps monitor the entire infrastructure and activate alarms, logs and events data. We used this resource to ingest logs from both of the EC2 machines to the CloudWatch stream;
- IAM roles, which are entities, similar to users, that can have access to services, depending on their level of permissions;
- Log groups, are previously configured log streams that enable the settings of monitorization;
- CloudWatch Agent Service, this agent enables the collection of system-level metrics, such as logs. With the agent active, a log stream is created in the log group;
- GuardDuty, is a threat detection service that monitors all of the existing accounts. With this service active, we were able to monitor any threat activity and deliver detailed security findings;
- Lambda (function), is a serverless compute service that allows the creation of simple expressions. A function was enabled to send an email whenever GuardDuty lists a finding on its threat monitoring activity;
- Sysmon, is a windows monitoring system that allowed us to control the detection of malicious activity;

3.2 Security Controls Implemented

- **Windows Server Hardening:** Hosted on a private subnet of the VPC and accessible via VPN tunneling. We created an OpenVPN server on the public subnet in order to access the resources (instances) on the private VPN. We configured a new AMI (Amazon Machine Image) to be able to run it and accessed its server to configure the account. Then, the machines were accessed via the private subnet through our machine, with the OpenVPN running as a bridge.
- **Linux Server Security Enhancements:** in order to secure the Linux server, we used the following script: https://github.com/fallen-man/ubuntu-22-04-cis-hardening (entrypoint.sh as the main script). We then installed “lynis” and ran a “sudo lynis audit system”.Afterwards, we checked the audit log for our hardening score.
- **Data Encryption:** For the Windows Server machine, we encrypted the data-at-rest using the Elastic Block Store service on AWS, which let us encrypt each data volume on each instance. Data was already encrypted-in-transit with OpenVPN, given the fact that this resource is a tunnel that’s encrypted at both ends and thus, the communication will be encrypted through it. For the Linux server instance, we downloaded a PII and PCI data file from: https://dlptest.com/sample-data/. The same process was made to encrypt data-at-rest and data-in-transit.
- **SIEM & Logging:** First we created an EC2 service linked-role in IAM. Then added our policy. We downloaded and installed the CloudWatch agent using the provided link in aws documentation (https://amazoncloudwatch-agent.s3.amazonaws.com/windows/amd64/latest/amazon-cloudwatch-agent.msi), then head over to Program Files > Amazon > AmazonCloudWatchAgent, create a text file and input this information to select our logs. The file was saved as “config.json” and ran the following command in Powershell: > cd 'C:/Program Files/Amazon/AmazonCloudWatchAgent/'; >./amazon-cloudwatch-agent-ctl.ps1 -a fetch-config -m ec2 -c file:config.json -s. This configuration enabled us to verify logs and log group/stream creation in CloudWatch.
- **Incident Detection & Response:** for the automation of the alerts, we enabled GuardDuty which let us monitor all the instances for threat activity. However, in order to receive these alerts, we created a Lambda function that sends an email whenever GuarDuty lists a finding on its threat monitoring activity.

3.3 Cloud Monitoring & Threat Detection

As mentioned before, CloudShield configured 2 instances, with CIS benchmarks, and installed Sysmon to generate detailed system logs. However, even though we intended to receive these logs, it was also important to limit the traffic that circulates in and out of those machines. For this reason, we implemented security groups, within the VPC, to control network flow.

The critical machines were placed inside a private subnet so that we could segment the network and separate the critical data. We also set up a GuardDuty aggregation and analyzed logs to provide findings about relevant ones. It was confirmed that GuardDuty was receiving all of the threat activity, but no alerts were being generated. To resolve this issue, we configured a Lambda function to automatically respond to threats and send an email every time an alert was generated.

---

4. Simulated Attack & Findings

4.1 Attack Scenario

The simulated attack was performed with a machine, placed in a public subnet, where it was executed a python script targeting a specific IP (13.48.95.79) that belongs to the Windows Server machine. The script enumerated the target machine and, then, performed an exploit by doing a brute force attack. The goal was to perform various steps, in the Mittre&attck, and show that the solution isn't just working on one form, but multiple.

Using one command (msfconsole -q -r resource_script.rc), we automated the process of using the script that will automatically exploit the machine and, afterwards, gain access to it. Here, a malicious eicar file was uploaded. After performing the attack, we analyzed the logs in GuardDuty.

This log analysis was possible due to the configured Lambda function, using a script, that allowed us to receive an alert (an email) every time a threat was received in GuardDuty.

4.2 Attack Execution & Detection

- **Tactics, Techniques, and Procedures (TTPs):** Mittre&Attck scripts, brute force attack, machine exploitation.

- **Security Event Log Analysis:** the attack was detected using a Lambda function (configured with a script)that would send an email whenever a new threat was received in GuardDuty.
- **Mitigation & Response:** in order to mitigate the threat, a list of blocked IP's was created.

---

5. Compliance & Risk Assessment

5.1 Compliance Frameworks Addressed

NIST (National Institute of Standards and Technology)

The NIST cybersecurity framework is a voluntary guidance for every company that needs help in organizing and improving their security sector. Titan Industrial Solutions specifically required for the project to be compliant with this framework. The following requirements were met:

- When the client, Titan Industrial Solutions, understood and assessed that their infrastructure needed to improve its security and mitigate possible threats, by sending an email to us (Governance);
- By detecting that the company operates a network of IoT-connected manufacturing facilities and relies on AWS to manage critical infrastructure. This can become a liability when AWS services are not well configured and sensitive data could be accessed by hackers (Identify);
- By hardening the system and implementing data encryption policy (Protect);
- By enabling resources (such as GuardDuty) to generate alarms everytime a new threat is detected (Detect);
- By immediately blocking and mitigate the threat (Respond);
- By taking action following the correct procedure (incident response plan) to recover from the threat (Recovery),

5.2 Risk Analysis & Security Recommendations

- **Identified Risks:** after the creation of the OpenVPN server/endpoint, the machines received several brute force attacks. If the machines were not correctly configured the attackers could have gained access.
- **Mitigation Strategies:** in order to block these IP's, a list was created to keep track of every single one of them.

---

6. Conclusion & Next Steps

With the implementation of several complex and detailed security measures, we were able to enhance and improve the company's cloud security infrastructure. The machines were hardened and their critical data storage was encrypted. A SIEM solution was also implemented, by using the GuardDuty has a log monitorization and alert service.

However, even though it is important to receive alerts about a threat, it is also important to have a mechanism that allows for those threats to be automatically mitigated. CloudShield team highly recommends an AWS integration that enables immediate response. For example, we recommend an automatic response every time GuardDuty lists a finding on its list, in order to mitigate the risk. With this automatization the threat actor can be immediately blocked from other attack attempts.

---

7. Supporting Documents & Appendices

- **AWS Cloud Architecture Diagram**
- **Security Incident Plan & SOPs**
- **Compliance Documentation**
- **Scripts & Configuration Files**

Prepared by:
CloudShield
Intensive Cybersecurity Bootcamp (ICC-01)